METHOD AND SYSTEM FOR CALCULATING RISK IN ASSOCIATION WITH A SECURITY AUDIT OF A COMPUTER NETWORK

PRIORITY AND RELATED APPLICATIONS

The present application claims priority to provisional patent application entitled, "Method and System for Configuring and Scheduling Security Audits of a Computer Network," filed on January 31, 2001 and assigned U.S. Application Serial Number 60/265,519. The present application also references and incorporates herein a related U.S. non-provisional patent application entitled, "Method and System for Configuring and Scheduling Security Audits of a Computer Network," filed concurrently herewith and having attorney docket number 05456.105009.

TECHNICAL FIELD

The present invention is generally directed to managing the security of a network. More specifically, the present invention facilitates computing a security score for elements in a distributed computing network.

BACKGROUND OF THE INVENTION

The security of computing networks is an increasingly important issue. With the growth of wide area networks (WANs), such as the Internet and the World Wide Web, people rely on computing networks to transfer and store an increasing amount of valuable information. This is also true of local area networks (LANs) used by companies, schools, organizations, and other enterprises. LANs typically are used by a bounded group of people in an organization to communicate and store electronic documents and information. LANs generally are coupled to or provide access to other local or wide area networks. Greater use and availability of computing networks produces a corresponding increase in the size and complexity of computing networks.

With the growth of networks and the importance of information available on the networks, there is also a need for better and more intelligent security. One approach to securing larger and more complex computer networks is to use a greater number and variety of security assessment devices. Security assessment devices can be used to evaluate elements in the network, such as desktop computers, servers, and routers, and to determine their respective

vulnerability to security problems, such as an attack from hackers. Security assessment devices can also be used more frequently to monitor the activity or status of the elements in a computing network. These network elements are commonly referred to as hosts. Throughout this specification the terms "host" and "element" will be used interchangeably to refer to the various components that can be found in a distributed computing network.

However, simply increasing the number of security assessment devices and the frequency with which they are used does not solve the problems presented in conventional network security. With increased security activity, a network administrator or other user must decide which elements in the network need to be audited, how frequently they should be audited, and what checks need to be run. These are decisions that often involve a variety of complicated factors and they are decisions that in practicality cannot be made every time a security audit is conducted. Increased assessment also produces a corresponding increase in the amount of security data that must be analyzed. A network administrator that is overwhelmed with security data is unable to make intelligent decisions about which security vulnerabilities should be addressed first.

An additional difficulty associated with maintaining adequate network security is finding the time to conduct security audits. Security audits generally must be initiated by a security professional and can hinder or entirely interrupt network performance for several hours at a time. These limitations place a premium on the time available to conduct security auditing and maintenance. Conventional network security systems do not support a means to accurately quantify security vulnerabilities so that they can be easily compared and prioritized.

In view of the foregoing, there is a need in the art for a system that will support the auditing of a distributed computing network. Specifically, a need exists to be able to automatically survey a network and prioritize any security issues identified by the survey. A further need exists to be able to assess the security risk of each element in the network. The assessment should reflect the importance of the element and, for each security vulnerability that exists on the element, the ease with which the vulnerability can be exploited, and the impact of exploiting the vulnerability. Moreover, a need exists to accurately quantify the risk posed by vulnerabilities so that they can be compared in association with a particular host and so that hosts can be compared over the entire network.

SUMMARY OF THE INVENTION

The present invention satisfies the above-described needs by providing a system and quantitative method for evaluating the security of elements in a network. A security audit system can collect data concerning elements in a network. This data can include the operating system and services running on the element and any vulnerabilities associated therewith. This information can be used to calculate a risk for each vulnerability associated with an element. Certain elements may have few vulnerabilities and other elements may have many vulnerabilities. In order to give each element a meaningful security score, a banded calculation method is used. The banded calculation method prevents many low-risk vulnerabilities associated with one element from overshadowing an element with a single high-risk vulnerability. This approach provides a simple means for a user to identify and address high-risk issues in a network.

In one aspect, the present invention comprises a method for computing a security score associated with a host in a distributed computing network. A security audit system can select a vulnerability identified in a host and obtain an asset value for the host. The asset value is typically assigned to the host based on its characteristics and functions. The security audit system can also retrieve an exploit probability and a severity value for the vulnerability. Security personnel generally consider the various types of vulnerabilities and select predetermined exploit probabilities and severity values. A risk value for a vulnerability can be computed from the host asset value, the exploit probability of the vulnerability, and the vulnerability's severity value. The risk value computation can be repeated for other vulnerabilities identified in the network. Because an element typically has multiple vulnerabilities, it is also useful to be able to compute a total security score for the element. The security audit system can use a banded calculation model to compute the total security score by placing the risk values in selected bands on a risk scale. The banded calculation model prevents several low risk values from being summed and producing a disproportionately and inaccurately large security score.

In another aspect, the present invention provides a method for computing a risk value for quantifying a vulnerability identified in a network. A network security system can receive an asset value for an element on which the vulnerability is detected. The asset value can be based on information collected during a security audit of the element. The network security system can

also receive a predetermined exploit probability and severity value for the vulnerability. Taking the asset value, the exploit probability, and the severity value, the network security system can compute a risk value that is useful in comparing other vulnerabilities in the network. The risk value can also be adjusted by a factor that reflects the difficulty of remedying the vulnerability.

These and other aspects of the invention will be described below in connection with the drawing set and the appended specification and claim set.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram illustrating an exemplary architecture for operating an embodiment of the present invention.

FIG. 2 is a logic flow diagram illustrating an overview of the operating steps performed by a security audit system in accordance with an exemplary embodiment of the present invention.

FIG. 3 is a logic flow diagram illustrating an exemplary process for analyzing the results of a security audit scan.

FIG. 4 is a logic flow diagram illustrating an exemplary process for calculating an asset value for a host.

FIG. 5 is a logic flow diagram illustrating an exemplary process for calculating a risk value for a vulnerability, and for accumulating these risk values for a single host.

FIG. 6 is a logic flow diagram illustrating an exemplary process for adjusting a vulnerability's calculated risk value as a function of the difficulty of fixing the vulnerability.

FIG. 7 is logic flow diagram illustrating an exemplary process for assigning a vulnerability's risk value to a risk band, and for incrementing the count of vulnerabilities assigned to the band.

FIG. 8A is a diagram illustrating a representative example of a banded scale useful for calculating a security score from one or more risk values.

FIG. 8B is a logic flow diagram illustrating an exemplary process for calculating a security score from the banded scale.

FIG. 9 is a logic flow diagram illustrating an exemplary process for determining the highest-risk band with at least one vulnerability risk value assigned to it.

FIG. 10 is a logic flow diagram illustrating an exemplary process for summing the risk values in each band on the banded scale.

DETAILED DESCRIPTION OF THE EXEMPLARY EMBODIMENTS

The present invention supports the assessment of the security risks of a computing network by providing a precise means to calculate and compare the risks posed by security vulnerabilities. Specifically, the present invention allows a security auditing system to identify security vulnerabilities in various elements throughout a network. The security auditing system also can collect information about the function and importance of elements in a computing network. Using this information, the invention calculates a risk value for each security vulnerability that is identified. The risk value can be prioritized based on the ease with which the vulnerability can be repaired. Prioritizing risk values for a particular network element assists a user or network administrator in deciding which vulnerabilities to address first. The invention also supports the calculation of a security score for a network element that accumulates the risk values of each vulnerability associated with the element. For example, employing a band calculation method ensures that a large number of low-risk vulnerabilities does not produce a higher security score than a smaller number of high-risk vulnerabilities. Calculating a security score for each element with the band calculation method allows for a more meaningful comparison of elements across a network.

Although the exemplary embodiments will be generally described in the context of software modules running in a distributed computing environment, those skilled in the art will recognize that the present invention also can be implemented in conjunction with other program modules for other types of computers. In a distributed computing environment, program modules may be physically located in different local and remote memory storage devices. Execution of the program modules may occur locally in a stand-alone manner or remotely in a client/server manner. Examples of such distributed computing environments include local area networks of an office, enterprise-wide computer networks, and the global Internet.

The detailed description that follows is represented largely in terms of processes and symbolic representations of operations in a distributed computing environment by conventional computer components, including database servers, application servers, mail servers, routers, security devices, firewalls, clients, workstations, memory storage devices, display devices, and

input devices. Each of these conventional distributed computing components is accessible via a communications network, such as a wide area network or local area network.

The processes and operations performed by the computer include the manipulation of signals by a client or server and the maintenance of these signals within data structures resident in one or more of the local or remote memory storage devices. Such data structures impose a physical organization upon the collection of data stored within a memory storage device and represent specific electrical or magnetic elements. These symbolic representations are the means used by those skilled in the art of computer programming and computer construction to most effectively convey teachings and discoveries to others skilled in the art.

The present invention also includes a computer program that embodies the functions described herein and illustrated in the appended flow charts. However, it should be apparent that there could be many different ways of implementing the invention in computer programming, and the invention should not be construed as limited to any one set of computer program instructions. Further, a skilled programmer would be able to write such a computer program to implement the disclosed invention based on the flow charts and associated description in the application text, for example. Therefore, disclosure of a particular set of program code instructions is not considered necessary for an adequate understanding of how to make and use the invention. The inventive functionality of the claimed computer program will be explained in more detail in the following description in conjunction with the remaining figures illustrating the program flow.

Referring now to the drawings, in which like numerals represent like elements throughout the several figures, aspects of the present invention and the preferred operating environment will be described.

Fig. 1 illustrates various aspects of an exemplary computing environment in which an embodiment of the present invention is designed to operate. Those skilled in the art will appreciate that Fig. 1 and the associated discussion are intended to provide a general description of representative computer network resources in an exemplary distributed computer environment including the inventive security audit system. The architecture comprises a console **105** and a security audit system **115** which are used to configure and schedule security audits of a network **110**. The console **105** communicates information about the current security state of the network **110** to a user. The console **105** typically comprises a graphical user interface for presenting and

managing data in a convenient format for the user. The console **105** is also operable for receiving information from the security audit system **115** and allowing control of the security audit system **115**. The security audit system **115** comprises an active scan engine **120** and one or more other scan engines. In the exemplary embodiment illustrated in Fig. 1, the active scan engine **120** is coupled to an Internet scanning engine **130**, a system scanning engine **150**, and a database scanning engine **140**. Each of these scan engines illustrated in Fig. 1 is coupled to a corresponding database.

The active scan engine's **120** primary task is acquiring and maintaining current data about the configuration and security posture of the network **110**. The active scan engine **120** utilizes the subsidiary scan engines **130**, **140** and **150** as a means for gathering information about the network **110**. The network **110** typically comprises elements such as desktop computers, routers, and various servers. The active scan engine **120** is responsible for coordinating the configuration, scheduling, and running of scans of these elements found in the network **110**. Typically, the active scan engine **120** is continuously running so that the scheduled scans can be run at their designated times, and the resultant data processed in a timely manner.

Referring to Fig. 2, an overview of an exemplary security auditing process **200** is shown. In alternative embodiments of the present invention, different auditing steps can be performed to collect information about a network that is used to compute a security score. In step **205**, an active scan engine **120** configures scans that are to be run on a network **110**. In step **210**, the active scan engine schedules the times at which the various configured scans will be run on the network **110**. An exemplary method for configuring and scheduling scans is described in greater detail in the U.S. non-provisional patent application filed concurrently herewith and referenced herein. When one of the scheduled times for running a scan is reached, the active scan engine **120** will run that scan on the network **110** in step **215**. A scan may be run against particular hosts in the network or over the entire network **110**. Typically, a scan collects information about the function of the hosts on the network and their respective vulnerabilities. This information is gathered, analyzed, and used to compute a security score for each host or element in the network in step **220**. The security scores assist a user or network administrator in determining which vulnerabilities should be addressed first.

Fig. 3 illustrates an exemplary process for analyzing the results of a security scan, as referenced in step **220**, for hosts located on the network **110**. In step **305**, the active scan engine

120 selects the ID of the first scan for which data are available. In step **310**, the active scan engine **120** selects the ID of the first host for which data are available in the current scan job results. In step **315**, the active scan engine **120** calculates an asset value for the current host. Asset values are calculated based on the operating system and services associated with the host. An asset value is a characteristic representing the importance of a particular host or element in the operation of the network. In step **320**, the active scan engine computes a risk value for each vulnerability detected on the current host. The risk value quantifies the risk a particular vulnerability presents for a network. As will be discussed in connection with Figs. 7 and 8A, each risk value calculated for a vulnerability is assigned to a band on the risk scale. Using the number of risk values assigned to each band, the active scan engine **120** computes a security score for the current host, in step **325**.

In step **330**, the active scan engine **120** determines if there are more hosts in the current job to be processed. If there are more hosts, the ID of the next host is retrieved in step **335**, and the process **220** returns to step **315**. In step **330**, if there are no more hosts to process in the current job, the active scan engine **120** determines if there are more scan jobs to process in step **340**. If there are more scan jobs, the ID of the next scan job is retrieved in step **340**, and the process returns to step **310**. In step **340**, if the active scan engine **120** finds that there are no more hosts to process, the analysis is complete.

Fig. 4 illustrates an exemplary method for calculating an asset value as referred to in step **315** of Fig. 3. Predetermined asset values are generally assigned for the various operating systems, host services, and the vulnerabilities that are likely to be encountered in a scan of a network. These predetermined asset values can be chosen by the provider of the security audit system or the user. The host's operating system and services are identified during the scan performed in step **215**. In step **405**, the asset value assigned to the host's particular operating system is retrieved. In steps **410** and **415**, respective asset values are retrieved for the host's services and relevant vulnerabilities. In the exemplary calculation illustrated herein, the highest of these three asset values is selected in step **420** and used as the asset value for the host in computing the risk values in Fig. 3. In alternative embodiments of the present invention, other processes, such as averaging the asset values, may be used to designate the significance of a particular host in a network **110**.

Fig. 5 illustrates an exemplary method for calculating risk values for the vulnerabilities associated with a host, as referred to in step **320** of Fig. 3. In step **505** the active scan engine **120** selects the first vulnerability discovered on the current host. In step **510**, the active scan engine **120** retrieves the asset value for the current host calculated in Fig. 4. The active scan engine **120** retrieves a value representing the probability of a successful exploit against the current vulnerability in step **515**. Each vulnerability has one or more exploit methods associated with it. The likelihood of a successful exploit is related to the difficulty of the exploit method. For example, certain vulnerabilities are easily exploited through the use of a simple, publicly available script. An easily exploited vulnerability is assigned a relatively high exploit probability value. On the other hand, other vulnerabilities are very difficult to exploit and involve a complex attack, such as the use of a buffer overflow. Because a buffer overflow is a difficult exploit method to implement, it is assigned a relatively low probability value. To retrieve a vulnerability's exploit probability value, the active scan engine **120** first retrieves the vulnerability's exploit method. The active scan engine **120** then consults a table that maps exploit methods to probability values. As outlined above, these values are assigned on the basis of the difficulty of each method. If a vulnerability has multiple exploit methods, the method with the highest exploit probability is selected.

In step **520**, the active scan engine **120** retrieves a predetermined severity value for the current vulnerability. This value represents the impact of a successful exploit of the vulnerability. In step **525**, the risk is calculated as the product of the host asset value and the vulnerability's severity and exploit probability values. Alternative embodiments of the present invention may weigh the three factors differently or use less than all three values in calculating the risk value. In step **530**, the active scan engine **120** adjusts the calculated risk value to take into account the difficulty of fixing the current vulnerability. The user can choose not to incorporate the fix difficulty factor and instead, use the "pure" risk value. In step **530**, the count of vulnerabilities assigned to the risk band that contains the adjusted risk value is incremented. In step **540**, the active scan engine **120** retrieves the next vulnerability detected on the current host. If there is such a vulnerability, the process **320** returns to step **515**. Otherwise, the calculation and accumulation of risk values for the host is complete.

Referring to Fig. 6, an exemplary process is illustrated for adjusting a risk value as referred to in step **530** of Fig. 5. In step **605**, the active scan engine **120** retrieves a fix difficulty

value for the vulnerability. The fix difficulty value is a measure of the difficulty of remedying a vulnerability. Each vulnerability has associated with it one or more fix methods. To retrieve a fix difficulty value for a vulnerability, the active scan engine **120** first retrieves the vulnerability's fix method. The active scan engine **120** then consults a table that maps each fix method to a difficulty value. This mapping table is typically created by network security personnel for use by the security audit system **115**. In step **610**, the risk value is multiplied by the fix difficulty value to yield an adjusted risk value. In step **615**, the adjusted risk value for the vulnerability can be compared to other adjusted risk values and ranked from highest to lowest. Using this exemplary method, when multiple vulnerabilities have the same computed risk, a user of the security audit system can address those that are easier to fix first. The adjusted risk value can also be used to calculate the security score so that an element's security score will reflect the difficulty with which its vulnerabilities can be fixed.

Fig. 7 illustrates an exemplary method for assigning a risk value to a risk band, and for incrementing the count of vulnerabilities assigned to the band, as referred to in step **535** of Fig. 5. A risk scale, as described in connection with Fig. 8, typically comprises several bands selected by the user. In step **705**, the active scan engine **120** identifies the band *i*, such that $r_i >= r(v) > r_{i-1}$, where r_i denotes the maximum risk value of band *i*, r_{i-1} denotes the maximum risk value of band *i-1*, and *r(v)* denotes the risk of the current vulnerability. In step **710**, the count C_i of vulnerabilities assigned to band *i* is incremented by 1.

Fig. 8A and Fig. 8B illustrate an exemplary method for calculating a security score as referred to in Step **325** of Fig. 3. The illustrated exemplary method employs a logarithmic band calculation so that security scores can be accurately compared over the entire network **110**. Using the logarithmic band calculation prevents a host with a large number of low risk values from achieving a higher security score than a host with a few higher risk values. In the illustration shown in Fig. 8A, risk values R_1, R_2, R_3, and R_4 are placed on a scale **800** that is divided into three bands. The number and width of the bands is arbitrary and can be varied to suit the needs and sensitivity of the network. The formula illustrated in Fig. 8A produces a security score that is a function of the band boundary values as opposed to the risk values. Using the exemplary risk values shown on the scale **800**, $r_{i(max)}$ is equal to 0.6, the upper boundary of the highest band containing a risk value. In the exemplary formula,

$$\text{Security Score} = r_{i(\max)-1} + \frac{r_{i(\max)} - r_{i(\max)-1}}{r_{i(\max)} + r_{i(\max)-1}} \sum_{k=1}^{\|R\|} \frac{r_{i(k)} + r_{i(k)-1}}{2^k}$$

the first term, $r_{i(max)\,-1}$, is 0.2, the lower boundary of the highest band containing a risk value. The remainder of the formula is a fraction of the middle band calculated from the band values encompassing the risk values placed on the scale **800**. The result of this calculation is that the security score will not exceed the upper boundary of the highest risk band, in this instance 0.6. The exemplary formula shown in Fig. 8A is merely one way of calculating a security score and alternative embodiments of the present invention may employ other formulas for the calculation.

Referring to Fig. 8B, an exemplary method for calculating a security score using the band calculation approach is illustrated. This exemplary method shows the calculation of a security score for a particular host in a network. This exemplary method can also be adapted to calculate a security score for a group of hosts within a network. Beginning with step **805**, the active scan engine **120** queries for the existence of vulnerabilities for the current host. If there are no vulnerabilities associated with the host, the security score is set to zero in step **810**. If there are vulnerabilities, the "yes" branch is followed to step **815**, where the active scan engine **120** determines the index of the highest-risk band that has at least one vulnerability's risk value assigned to it. As illustrated in Fig. 8A, the lowest band contains one risk value and the second band contains three risk values. In step **815**, the highest risk band which contains at least one risk value is identified as band $b_{i(max)}$. In the example shown in Fig. 8A, band $b_{i(max)}$ is the middle band bounded by 0.2 and 0.6 on the risk scale. As noted in Fig. 8A, an embodiment of the current invention implements a simplified version of the band calculation formula. This simplified calculation is accomplished in steps **820** and **825**. In step **820**, the simplified summation calculation yields preliminary score *s*. In step **825**, the security score is computed by multiplying the summation value by the fraction $(r_{i(max)} - r_{i(max)-1})/(r_{i(max)} + r_{i(max)-1})$, and adding $r_{i(max)-1}$, as noted in the formula above. In alternative embodiments of the present invention other methods may be used to sum the risk values to get an accumulated security score.

Fig. 9 illustrates an exemplary method for determining the highest-risk band in which a risk value is present. In the example shown in Fig. 8A, the second band is band $b_{i(max)}$ because there are no risk values in the upper region between 0.6 and 1.0. In step **905**, the index is set to the highest band risk. In step **910**, if the index is equal to b_{min}, the lowest risk band, the process

ends. Otherwise, in step **915** the active scan engine **120** determines if at least one risk value has been assigned to the current band. If so, the current band index is the desired one and the process is complete. Otherwise, the current index is decremented in step **920** and the process **815** returns to step **910.**

Fig. 10 illustrates an exemplary process for summing the banded vulnerability risk values as referenced in step **820**. In step **1005**, the calculation is initialized by setting the current band index *i* equal to $b_{i(max)}$. The preliminary security score *s* is set to 0.0. The value *PreviousCount*, which represents the number of risk values that have already been processed, is set to 0. The value *StartIndex*, which represents the index of the first risk value in the current band, is set to 1. In step **1010**, if the current band is band b_{min}, the process is complete. Otherwise, control passes to step **1015**, where a value *n* is set equal to the count of risks C_i assigned to the current band plus the value *PreviousCount*. The preliminary score *s* is incremented by $(r_i + r_{i-1}) * (2^{n-StartIndex+1} - 1.0)/2^n$. The values *StartIndex* and *PreviousCount* are then incremented by C_i. The current band index *i* is then decremented by 1, and the process **820** returns to step **1010**.

In conclusion, the present invention enables and supports security auditing of a distributed computing network by providing a useful numerical value of the risk associated with a vulnerability or group of vulnerabilities. The security audit system can collect information about the elements of a network and compute a risk value for vulnerabilities detected therein. The risk value can be based on the importance of the network element, the likelihood of exploit, and the potential for damage to the network in the event the vulnerability is exploited. The risk value can also be adjusted to reflect the difficulty of remedying the vulnerability. The security audit system can also collect risk values for a particular element and compute a total security score for the element. The security audit system uses a banded calculation method to ensure that a host with several low risk values does not have a higher security score than a host with a few high risk values. Security scores are useful for comparing individual elements or groups of elements on the network.

It will be appreciated that the present invention fulfills the needs of the prior art described herein and meets the above-stated objects. While there has been shown and described the preferred embodiment of the invention, it will be evident to those skilled in the art that various

modifications and changes may be made thereto without departing from the spirit and the scope of the invention as set forth in the appended claims and equivalence thereof. Although the present invention has been described as operating on a local area network, it should be understood that the invention can be applied to other types of distributed computing environments. Furthermore, it should be readily apparent that portions of the calculation can be varied in order modify the results without departing from the scope of the invention.